## NAME

Maq – Mapping and Assembly with Qualities

## SYNOPSIS

**maq** *command* [*options*] *arguments*

**maq.pl** *command* [*options*] *arguments*

## DESCRIPTION

Maq is a software that builds mapping assemblies from short reads generated by the next-generation sequencing machines. It is particularly designed for Illumina-Solexa 1G Genetic Analyzer, and has a preliminary functionality to handle AB SOLiD data.

With Maq you can:

- Fast align Illumina/SOLiD reads to the reference genome. With the default options, one million pairs of reads can be mapped to the human genome in about 10 CPU hours with less than 1G memory.
- Accurately measure the error probability of the alignment of each individual read.
- Call the consensus genotypes, including homozygous and heterozygous polymorphisms, with a Phred probabilistic quality assigned to each base.
- Find short indels with paired end reads.
- Accurately find large scale genomic deletions and translocations with paired end reads.
- Discover potential CNVs by checking read depth.
- Evaluate the accuracy of raw base qualities from sequencers and help to check the systematic errors.

However, Maq can **NOT**:

- Do *de novo* assembly. (Maq can only call the consensus by mapping reads to a known reference.)
- Map shorts reads against themselves. (Maq can only find complete overlap between reads.)
- Align capillary reads or 454 reads to the reference. (Maq cannot align reads longer than 63bp.)

## MAQ COMMANDS

### Key Commands

**fasta2bfa** **maq fasta2bfa** *in.ref.fasta out.ref.bfa*

Convert sequences in FASTA format to Maq’s BFA (binary FASTA) format.

**fastq2bfq** **maq fastq2bfq** [**−n** *nreads*] *in.read.fastq out.read.bfq*|*out.prefix*

Convert reads in FASTQ format to Maq’s BFQ (binary FASTQ) format.

**OPTIONS:**

| | |
|---|---|
| **−n** *INT* | number of reads per file [not specified] |

**map** **maq map** [**−n** *nmis*] [**−a** *maxins*] [**−c**] [**−1** *len1*] [**−2** *len2*] [**−d** *adap3*] [**−m** *mutrate*] [**−u** *unmapped*] [−e *maxerr*] [**−M** c|g] [**−N**] [**−H** *allhits*] [**−C** *maxhits*] *out.aln.map in.ref.bfa in.read1.bfq* [*in.read2.bfq*] 2> *out.map.log*

Map reads to the reference sequences.

**OPTIONS:**

| | |
|---|---|
| **−n** *INT* | Number of maximum mismatches that can always be found [2] |
| **−a** *INT* | Maximum outer distance for a correct read pair [250] |
| **−A** *INT* | Maximum outer distance of two RF paied read (0 for disable) [0] |
| **−c** | Map reads in the colour space (for SOLiD only) |
| **−1** *INT* | Read length for the first read, 0 for auto [0] |

| | |
|---|---|
| **−2** *INT* | Read length for the second read, 0 for auto [0] |
| **−m** *FLOAT* | Mutation rate between the reference sequences and the reads [0.001] |
| **−d** *FILE* | Specify a file containing a single line of the 3’−adapter sequence [null] |
| **−u** *FILE* | Dump unmapped reads and reads containing more than *nmis* mismatches to a separate file [null] |
| **−e** *INT* | Threshold on the sum of mismatching base qualities [70] |
| **−H** *FILE* | Dump multiple/all 01−mismatch hits to *FILE* [null] |
| **−C** *INT* | Maximum number of hits to output. Unlimited if larger than 512. [250] |
| **−M** c\|g | methylation alignment mode. All C (or G) on the forward strand will be changed to T (or A). This option is for testing only. |
| **−N** | store the mismatch position in the output file *out.aln.map*. When this option is in use, the maximum allowed read length is 55bp. |

**NOTE:**

* Paired end reads should be prepared in two files, one for each end, with reads are sorted in the same order. This means the k−th read in the first file is mated with the k−th read in the second file. The corresponding read names must be identical up to the tailing ‘/1’ or ‘/2’. For example, such a pair of read names are allowed: ‘EAS1_1_5_100_200/1’ and ‘EAS1_1_5_100_200/2’. The tailing ‘/[12]’ is usually generated by the GAPipeline to distinguish the two ends in a pair.

* The output is a compressed binary file. It is affected by the endianness.

* The best way to run this command is to provide about 1 to 3 million reads as input. More reads consume more memory.

* Option **−n** controls the sensitivity of the alignment. By default, a hit with up to 2 mismatches can be always found. Higher **−n** finds more hits and also improves the accuracy of mapping qualities. However, this is done at the cost of speed.

* Alignments with many high-quality mismatches should be discarded as false alignments or possible contaminations. This behaviour is controlled by option **−e**. The **−e** threshold is only calculated approximately because base qualities are divided by 10 at a certain stage of the alignment. The **−Q** option in the **assemble** command precisely set the threshold.

* A pair of reads are said to be correctly paired if and only if the orientation is *FR* and the outer distance of the pair is no larger than *maxins*. There is no limit on the minimum insert size. This setting is determined by the paired end alignment algorithm used in Maq. Requiring a minimum insert size will lead to some wrong alignments with highly overestimated mapping qualities.

* Currently, read pairs from Illumina/Solexa long-insert library have RF read orientation. The maximum insert size is set by option **−A**. However, long-insert library is also mixed with a small fraction of short-insert read pairs. **−a** should also be set correctly.

* Sometimes 5’−end or even the entire 3’−adapter sequence may be sequenced. Providing **−d** renders Maq to eliminate the adapter contaminations.

* Given 2 million reads as input, **maq** usually takes 800MB memory.

**mapmerge** **maq mapmerge** *out.aln.map in.aln1.map in.aln2.map* [...]

Merge a batch of read alignments together.

**NOTE:**

- * In theory, this command can merge unlimited number of alignments. However, as mapmerge will be reading all the inputs at the same time, it may hit the limit of the maximum number of opening files set by the OS. At present, this has to be manually solved by endusers.
- * Command **mapmerge** can be used to merge alignment files with different read lengths. All the subsequent analyses do not assume fixed length any more.

**rmdup** **maq rmdup** *out.rmdup.map in.ori.map*

Remove pairs with identical outer coordinates. In principle, pairs with identical outer coordinates should happen rarely. However, due to the amplification in sample preparation, this occurs much more frequently than by chance. Practical analyses show that removing duplicates helps to improve the overall accuracy of SNP calling.

**assemble** **maq assemble** [**−sp**] [**−m** *maxmis*] [**−Q** *maxerr*] [**−r** *hetrate*] [**−t** *coef*] [**−q** *minQ*] [**−N** *nHap*] *out.cns in.ref.bfa in.aln.map* 2> *out.cns.log*

Call the consensus sequences from read mapping.

**OPTIONS:**

| | |
|---|---|
| **−t** *FLOAT* | Error dependency coefficient [0.93] |
| **−r** *FLOAT* | Fraction of heterozygotes among all sites [0.001] |
| **−s** | Take single end mapping quality as the final mapping quality; otherwise paired end mapping quality will be used |
| **−p** | Discard paired end reads that are not mapped in correct pairs |
| **−m** *INT* | Maximum number of mismatches allowed for a read to be used in consensus calling [7] |
| **−Q** *INT* | Maximum allowed sum of quality values of mismatched bases [60] |
| **−q** *INT* | Minimum mapping quality allowed for a read to be used in consensus calling [0] |
| **−N** *INT* | Number of haplotypes in the pool (>=2) [2] |

**NOTE:**

- * Option **−Q** sets a limit on the maximum sum of mismatching base qualities. Reads containing many high-quality mismatches should be discarded.
- * Option **−N** sets the number of haplotypes in a pool. It is designed for resequencing of samples by pooling multiple strains/individuals together. For diploid genome resequencing, this option equals 2.

**glfgen** **maq glfgen** [**−sp**] [**−m** *maxmis*] [**−Q** *maxerr*] [**−r** *hetrate*] [**−t** *coef*] [**−q** *minQ*] [**−N** *nHap*] *out.cns in.ref.bfa in.aln.map* 2> *out.cns.log*

Calculate log-likelihood for all genotypes and store the results in GLF format (Genotyping Likelihood Format). Please check MAQ website for detailed descriptions of the file format and the related utilities.

**indelpe** **maq indelpe** *in.ref.bfa in.aln.map* > *out.indelpe*

Call consistent indels from paired end reads. The output is TAB delimited with each line consisting of chromosome, start position, type of the indel, number of reads across the indel, size of the indel and inserted/deleted nucleotides (separated by colon), number of indels on the reverse strand, number of indels on the forward strand, 5' sequence ahead of the indel, 3' sequence following the indel, number of reads aligned without indels and three additional columns for filters.

At the 3rd column, type of the indel, a star indicates the indel is confirmed by reads from both strands, a plus means the indel is hit by at least two reads but from the same strand, a minus shows the indel is only found on one read, and a dot means the indel is too close to another indel and is filtered out.

Users are recommended to run through ‘maq.pl indelpe’ to correct the number of reads mapped without indels. For more details, see the ‘maq.pl indelpe’ section.

**indelsoa** **maq indelsoa** *in.ref.bfa in.aln.map* > *out.indelsoa*

Call potential homozygous indels and break points by detecting the abnormal alignment pattern around indels and break points. The output is also TAB delimited with each line consisting of chromosome, approximate coordinate, length of the abnormal region, number of reads mapped across the position, number of reads on the left-hand side of the position and number of reads on the right-hand side. The last column can be ignored.

The output contains many false positives. A recommended filter could be:

```
awk '$5+$6-$4 >= 3 && $4 <= 1' in.indelsoa
```

Note that this command does not aim to be an accurate indel detector, but mainly helps to avoid some false positives in substitution calling. In addition, it only works well given deep depth (~40X for example); otherwise the false negative rate would be very high.

**Format Converting**

**sol2sanger** **maq sol2sanger** *in.sol.fastq out.sanger.fastq*

Convert Solexa FASTQ to standard/Sanger FASTQ format.

**bfq2fastq** **maq bfq2fastq** *in.read.bfq out.read.fastq*

Convert Maq’s BFQ format to standard FASTQ format.

**mapass2maq**

**maq mapass2maq** *in.mapass2.map out.maq.map*

Convert obsolete mapass2’s map format to Maq’s map format. The old format does not contain read names.

**Information Extracting**

**mapview** **maq mapview** [**−bN**] *in.aln.map* > *out.aln.txt*

Display the read alignment in plain text. For reads aligned before the Smith-Waterman alignment, each line consists of read name, chromosome, position, strand, insert size from the outer coorniates of a pair, paired flag, mapping quality, single-end mapping quality, alternative mapping quality, number of mismatches of the best hit, sum of qualities of mismatched bases of the best hit, number of 0−mismatch hits of the first 24bp, number of 1−mismatch hits of the first 24bp on the reference, length of the read, read sequence and its quality. Alternative mapping quality always equals to mapping quality if the reads are not paired. If reads are paired, it equals to the smaller mapping quality of the two ends. This alternative mapping quality is actually the mapping quality of an abnormal pair.

The fifth column, paired flag, is a bitwise flag. Its lower 4 bits give the orientation: 1 stands for FF, 2 for FR, 4 for RF, and 8 for RR, where FR means that the read with smaller coordinate is on the forward strand, and its mate is on the reverse strand. Only FR is allowed for a correct pair. The higher bits of this flag give further information. If the pair meets the paired end requirement, 16 will be set. If the two reads are mapped to different chromosomes, 32 will be set. If one of the two reads cannot be mapped at all, 64 will be set. The flag for a correct pair always equals to 18.

For reads aligned by the Smith-Waterman alignment afterwards, the flag is always 130. A line consists of read name, chromosome, position, strand, insert size, flag (always 130), position of the indel on the read (0 if no indel), length of the indels (positive for insertions and negative for deletions), mapping quality of its mate, number of mismatches of the best hit, sum of qualities of mismatched bases of the best hit, two zeros, length of the read, read sequence and its quality. The mate of a 130−flagged read always gets a flag 18.

Flag 192 indicates that the read is not mapped but its mate is mapped. For such a read pair, one read has flag 64 and the other has 192.

**OPTIONS:**

| | |
|---|---|
| **−b** | do not display the read sequence and the quality |
| **−N** | display the positions where mismatches occur. This flag only works with a .map file generated by ‘maq map −N’. |

**mapcheck** **maq mapcheck** [**−s**] [**−m** *maxmis*] [**−q** *minQ*] *in.ref.bfa in.aln.map* > *out.mapcheck*

Read quality check. The mapcheck first reports the composition and the depth of the reference. After that there is a form. The first column indicates the position on a read. Following four columns which show the nucleotide composition, substitution rates between the reference and reads will be given. These rates and the numbers in the following columns are scaled to 999 and rounded to nearest integer. The next group of columns show the distribution of base qualities along the reads at a quality interval of 10. A decay in quality can usually be observed, which means bases at the end of read are less accurate. The last group of columns present the fraction of substitutions for read bases at a quality interval. This measures the accuracy of base quality estimation. Idealy, we expect to see 1 in the 3? column, 10 in the 2? column and 100 in the 1? column.

**OPTIONS:**

| | |
|---|---|
| **−s** | Take single end mapping quality as the final mapping quality |
| **−m** *INT* | Maximum number of mismatahces allowed for a read to be counted [4] |
| **−q** *INT* | Minimum mapping quality allowed for a read to be counted [30] |

**pileup** **maq pileup** [**−spvP**] [**−m** *maxmis*] [**−Q** *maxerr*] [**−q** *minQ*] [**−l** *sitefile*] *in.ref.bfa in.aln.map* > *out.pileup*

Display the alignment in a ‘pileup’ text format. Each line consists of chromosome, position, reference base, depth and the bases on reads that cover this position. If **−v** is added on the command line, base qualities and mapping qualities will be presented in the sixth and seventh columns in order.

The fifth column always starts with ‘@’. In this column, read bases identical to the reference are showed in comma ‘,’ or dot ‘.’, and read bases different from the reference in letters. A comma or a upper case indicates that the base comes from a read aligned on the forward strand, while a dot or a lower case on the reverse strand.

This command is for users who want to develop their own SNP callers.

**OPTIONS:**

| | |
|---|---|
| **−s** | Take single end mapping quality as the final mapping quality |
| **−p** | Discard paired end reads that are not mapped as correct pairs |
| **−v** | Output verbose information including base qualities and mapping qualities |
| **−m** *INT* | Maximum number of mismatches allowed for a read to be used [7] |
| **−Q** *INT* | Maximum allowed number of quality values of mismatches [60] |
| **−q** *INT* | Minimum mapping quality allowed for a read to be used [0] |
| **−l** *FILE* | File containing the sites at which pileup will be printed out. In this file the first column gives the names of the reference and the second the coordinates. Additional |

columns will be ignored. [null]

| | |
|---|---|
| **−P** | also output the base position on the read |

**cns2fq** **maq cns2fq** [**−Q** *minMapQ*] [**−n** *minNeiQ*] [**−d** *minDepth*] [**−D** *maxDepth*] *in.cns* > *out.cns.fastq*

Extract the consensus sequences in FASTQ format. In the sequence lines, bases in lower case are essentially repeats or do not have sufficient coverage; bases in upper case indicate regions where SNPs can be reliably called. In the quality lines, ASCII of a character minus 33 gives the PHRED quality.

**OPTIONS:**

| | |
|---|---|
| **−Q** *INT* | Minimum mapping quality [40] |
| **−d** *INT* | Minimum read depth [3] |
| **−n** *INT* | Minimum neighbouring quality [20] |
| **−D** *INT* | Maximum read dpeth. >=255 for unlimited. [255] |

**cns2snp** **maq cns2snp** *in.cns* > *out.snp*

Extract SNP sites. Each line consists of chromosome, position, reference base, consensus base, Phred-like consensus quality, read depth, the average number of hits of reads covering this position, the highest mapping quality of the reads covering the position, the minimum consensus quality in the 3bp flanking regions at each side of the site (6bp in total), the second best call, log likelihood ratio of the second best and the third best call, and the third best call.

The 5th column is the key criterion when you judge the reliability of a SNP. However, as this quality is only calculated assuming site independency, you should also consider other columns to get more accurate SNP calls. Script command '**maq.pl SNPfilter**' is designed for this (see below).

The 7th column implies whether the site falls in a repetitive region. If no read covering the site can be mapped with high mapping quality, the flanking region is possibly repetitive or in the lack of good reads. A SNP at such site is usually not reliable.

The 8th column roughly gives the copy number of the flanking region in the reference genome. In most cases, this number approaches 1.00, which means the region is about unique. Sometimes you may see non-zero read depth but 0.00 at the 7th column. This indicates that all the reads covering the position have at least two mismatches. Maq only counts the number of 0− and 1−mismatch hits to the reference. This is due to a complex technical issue.

The 9th column gives the neighbouring quality. Filtering on this column is also required to get reliable SNPs. This idea is inspired by NQS, although NQS is initially designed for a single read instead of a consensus.

**cns2view** **maq cns2view** *in.cns* > *out.view*

Show detailed information at all sites. The output format is identical to **cns2snp** report.

**cns2ref** **maq cns2ref** *in.cns* > *out.ref.fasta*

Extract the reference sequence.

**cns2win** **maq cns2win** [**−w** *winsize*] [**−c** *chr*] [**−b** *begin*] [**−e** *end*] [**−q** *minQ*] *in.cns* > *out.win*

Extract information averaged in a tilling window. The output is TAB delimited, which consists of reference name, coordinate divided by 1,000,000, SNP rate, het rate, raw read depth, read depth in approximately unique regions, the average number of hits of reads in the window and percent GC.

**OPTIONS:**

| | |
|---|---|
| **−w** *INT* | Size of a window [1000] |
| **−c** *STR* | Destinated reference sequence; otherwise all references will be used [null] |
| **−b** *INT* | Start position, 0 for no constraint [0] |
| **−e** *INT* | End position, 0 for no constraint [0] |
| **−q** *INT* | Minimum consensus quality of the sites to be used [0] |

### Simulation Related

**fakemut** **maq fakemut** [**−r** *mutrate*] [**−R** *indelfrac*] *in.ref.fasta* > *out.fakeref.fasta* 2> *out.fake.snp*

Randomly introduce substitutions and indels to the reference. Substitutions and sinlge basepair indels can be added.

**OPTIONS:**

| | |
|---|---|
| **−r** *FLOAT* | Mutation rate [0.001] |
| **−R** *FLOAT* | Fraction of mutations to be indels [0.1] |

**simutrain** **maq simutrain** *out.simupars.dat in.read.fastq*

Estimate/train parameters for read simulation.

**simulate** **maq simulate** [**−d** *insize*] [**−s** *stdev*] [**−N** *nReads*] [**−1** *readLen1*] [**−2** *readLen2*] [**−r** *mutRate*] [**−R** *indelFrac*] [**−h**] *out.read1.fastq out.read2.fastq in.ref.fasta in.simupars.dat*

Simulate paired end reads. File *in.simupars.dat* determines the read lengths and quality distribution. It is generated from **simutrain**, or can be downloaded from Maq website. In the output read files, a read name consists of the reference sequence name and the outer coordinates of the pair of simulated reads. By default, **simulate** assumes reads come from a diploid sequence which is generated by adding two different sets of mutations, including one basepair indels, to *in.ref.fasta*.

**OPTIONS:**

| | |
|---|---|
| **−d** *INT* | mean of the outer distance of insert sizes [170] |
| **−s** *INT* | standard deviation of insert sizes [20] |
| **−N** *INT* | number of pairs of reads to be generated [1000000] |
| **−1** *INT* | length of the first read [set by *in.simupars.dat*] |
| **−2** *INT* | length of the second read [set by *in.simupars.dat*] |
| **−r** *FLOAT* | mutation rate [0.001] |
| **−R** *FLOAT* | fraction of 1bp indels [0.1] |
| **−h** | add all mutations to *in.ref.fasta* and generate reads from the single mutated sequence (haploid mode) |

**NOTE:**

* Reads generated from this command are independent, which deviates from the truth. Whereas alignment evaluation is less affected by this, evaluation on SNP calling should be performed with caution. Error dependency may be one of the major causes of wrong SNP calls.

**simustat** **maq simustat** *in.simu−aln.map* > *out.simustat*

Evaluate mapping qualities from simulated reads.

### SOLiD Related

**fasta2csfa** **maq fasta2csfa** *in.nucl−ref.fasta* > *out.colour−ref.fasta*

Convert nucleotide FASTA to colour-coded FASTA. Flag **−c** should be then applied to **map** command. In the output, letter 'A' stands for color 0, 'C' for 1, 'G' for 2 and 'T' for 3. Each sequence in the output is 1bp shorter than the input.

**csmap2nt** **maq csmap2nt** *out.nt.map in.ref.nt.bfa in.cs.map*

Convert color alignment to nucleotide alignment. The input *in.ref.nt.bfa* is the nucleotide binary FASTA reference file. It must correspond to the original file from which the color reference is converted. Nucleotide consensus can be called from the resultant alignment.

### Miscellaneous/Advanced Commands

**submap** **maq submap** [**−q** *minMapQ*] [**−Q** *maxSumErr*] [**−m** *maxMM*] [**−p**] *out.map in.map*

Filter bad alignments in *in.map*. Command-line options are described in the '**assemble**' command.

**eland2maq** **maq eland2maq** [**−q** *defqual*] *out.map in.list in.eland*

Convert eland alignment to maq's .map format. File *in.list* consists of the sequence names that appear at the seventh column of the eland alignment file *in.eland* and the name you expect to see in maq alignment. The following is an example:

```
cX.fa chrX
c1.fa chr1
c2.fa chr2
```

If you are aligning reads in several batches using eland, it is important to use the same *in.list* for the conversion. In addition, maq will load all the alignments and sort them in the memory. If you have concatenate several eland outputs into one huge file, you should separate it into smaller files to prevent maq from eating all your machine memory.

This command actually aims to show Eland alignment in Maqview. As no quality information is available, the resultant maq alignment file should not be used to call consensus genotypes.

**export2maq**

**maq export2maq** [**−1** *read1len*] [**−2** *read2len*] [**−a** *maxdist*] [**−n**] *out.map in.list in.export*

Convert Illumina's Export format to Maq's *.map* format. Export format is a new alignment format since SolexaPipeline−0.3.0 which also calculates mapping qualities like maq. The resultant file can be used to call consensus genotypes as most of necessary information is available for maq to do this accurately.

**OPTIONS:**

| | |
|---|---|
| **−1** *INT* | Length of the first read [0] |
| **−2** *INT* | Length of the second read [0] |
| **−a** *INT* | Maximum outer distance for a correct read pair [250] |
| **−n** | Retain filtered reads |

## MAQ-PERL COMMANDS

**demo** **maq.pl demo** [**−h**] [**−s**] [**−N** *nPairs*] [**−d** *outDir*] *in.fasta in.simudat*

Demonstrate the use of **maq** and its companion scripts. This command will simulate reads from a FASTA file *in.fasta*. The sequence length and qualities are determined by *in.simudat* which is generated from **maq simutrain** or can be downloaded from Maq website. The simulated reads will then be mapped with **maq.pl easyrun**. The alignment accuracy is evaluated by **maq simustat**, the consensus accuracy by **maq simucns**, and the SNP accuracy by

**maq_eval.pl**.

By default, paired end reads will be simulated and a diploid sequence will be generated from the input by adding mutations to either haploid type. The insert size and mutation rate are controlled by **maq simulate**.

**OPTIONS:**

| | |
|---|---|
| **–h** | simulate a haploid sequence instead of a diploid sequence |
| **–s** | use single-end mode to align reads instead of paired-end mode |
| **–N** *INT* | number of pairs of reads to be simulated [1000000] |
| **–d** *DIR* | output directory [maqdemo] |

**NOTE:**

* The output files from **maq_eval.pl** have not been documented, but you may make a good guess at some of these files.

* This command just demonstrates the use of the maq suite. The accuracy on real data is almost always lower than what you see from pure simulation.

**easyrun** **maq.pl easyrun** [**–1** *read1Len*] [**–d** *out.dir*] [**–n** *nReads*] [**–A** *3adapter*] [**–e** *minDep*] [**–q** *minCnsQ*] [**–p**] [**–2** *read2Len*] [**–a** *maxIns*] [**–S**] [**–N**] *in.ref.fasta in1.fastq* [*in2.fastq*]

Analyses pipeline for small genomes. Easyrun command will run most of analyses implemented in **maq**. By default, **easyrun** assumes all the input read sequences files are single-end and independent; when **–p** is specified, two read sequence files are required, one for each end.

Several files will be generated in *out.dir*, among which the following files are the key output:

| | |
|---|---|
| *cns.final.snp* | final SNP calls with low quality ones filtered out |
| *cns.fq* | consensus sequences and qualities in the FASTQ format |

**OPTIONS:**

| | |
|---|---|
| **–d** *DIR* | output directory [easyrun] |
| **–n** *INT* | number of reads/pairs in one batch of alignment [2000000] |
| **–S** | apply split-read analysis of short indels (maybe very slow) |
| **–N** *INT* | number of haplotypes/strains in the pool (>=2) [2] |
| **–A** *FILE* | file for 3'–adapter. The file should contain a single line of sequence [null] |
| **–1** *INT* | length of the first read, 0 for auto [0] |
| **–e** *INT* | minimum read depth required to call a SNP (for SNPfilter) [3] |
| **–q** *INT* | minimum consensus quality for SNPs in *cns.final.snp* [30] |
| **–p** | switch to paired end alignment mode |
| **–2** *INT* | length of the second read when **–p** is applied [0] |
| **–a** *INT* | maximum insert size when **–p** is applied [250] |

**NOTES:**

* For SNP calling on pooled samples, users should set correct '**–N**' as well as '**–E** 0'.

* The input file can be maq's binary format. **maq.pl** will automatically detect the file format.

**SNPfilter** **maq.pl SNPfilter** [**–d** *minDep*] [**–D** *maxDep*] [**–Q** *maxMapQ*] [**–q** *minCnsQ*] [**–w** *indelWinSize*] [**–n** *minNeiQ*] [**–F** *in.indelpe*] [**–f** *in.indelsoa*] [**–s** *minScore*] [**–m** *maxAcross*] [**–a**] [**–N** *maxWinSNP*] [**–W** *densWinSize*] *in.cns2snp.snp* > *out.filtered.snp*

Rule out SNPs that are covered by few reads (specified by **–d**), by too many reads (specified by **–D**), near (specified by **–w**) to a potential indel, falling in a possible repetitve region (characterized by **–Q**), or having low-quality neighbouring bases (specified by **–n**). If *maxWinSNP* or more SNPs appear in any *densWinSize* window, they will also be filtered out together.

**OPTIONS:**

| | |
|---|---|
| **–d** *INT* | Minimum read depth required to call a SNP [3] |
| **–D** *INT* | Maximum read depth required to call a SNP (<255, otherwise ignored) [256] |
| **–Q** *INT* | Required maximum mapping quality of reads covering the SNP [40] |
| **–q** *INT* | Minimum consensus quality [20] |
| **–n** *INT* | Minimum adjacent consensus quality [20] |
| **–w** *INT* | Size of the window around the potential indels. SNPs that are close to indels will be suppressed [3] |
| **–F** *FILE* | The **indelpe** output [null] |
| **–f** *FILE* | The **indelsoa** output [null] |
| **–s** *INT* | Minimum score for a soa-indel to be considered [3] |
| **–m** *INT* | Maximum number of reads that can be mapped across a soa-indel [1] |
| **–a** | Alternative filter for single end alignment |

**indelpe** **maq.pl indelpe** *in.indelpe* > *out.indelpe*

Correct the number of reads mapped without indels for homopolymer tracts. This command modify the 4th, 10th and the last three columns of *in.indelpe* and output the result in *out.indelpe*. After the correction, the following **awk** command gives putative homozygous indels:

```
awk '($3=="*"||$3=="+") && $6+$7>=3 && ($6+$7)/$4>=0.75'
```

and the following gives heterozygotes:

```
awk '($3=="*"||$3=="+") && $6+$7>=3 && ($6+$7)/$4<0.75'
```

Please note that this **indelpe** command just implements several heuristic rules. It does not correct for impure homopolymer runs or di–nucleotide/triplet repeats. Consequently, the two awk commands only give approximate hom/het indels.

## EXAMPLES

- Easyrun script:

  ```
  maq.pl easyrun –d easyrun ref.fasta part1.fastq part2.fastq
  ```

- Key commands behind easyrun:

  ```
  maq fasta2bfa ref.fasta ref.bfa;
  maq fastq2bfq part1.fastq part1.bfq;
  maq fastq2bfq part2.fastq part2.bfq;
  maq map part1.map ref.bfa part1.bfq;
  maq map part2.map ref.bfa part2.bfq;
  maq mapmerge aln.map part1.map part2.map;
  maq assemble cns.cns ref.bfa aln.map;
  ```

## LICENSE

GNU General Public License, version 3 (GPLv3)

## AVAILABILITY

<http://maq.sourceforge.net>

## AUTHOR

Heng Li <lh3@sanger.ac.uk>